Архитектура операционной системы

Ядро и вспомогательные модули операционной системы

Многослойная архитектура операционной системы

- Большинство современных операционных систем представляют собой хорошо **структурированные модульные системы**, в которые изначально заложены способности к дальнейшему развитию, расширению и переносу на новые платформы. Какой – либо единой архитектуры операционных систем не существует, но существуют универсальные подходы к структурированию операционных систем.

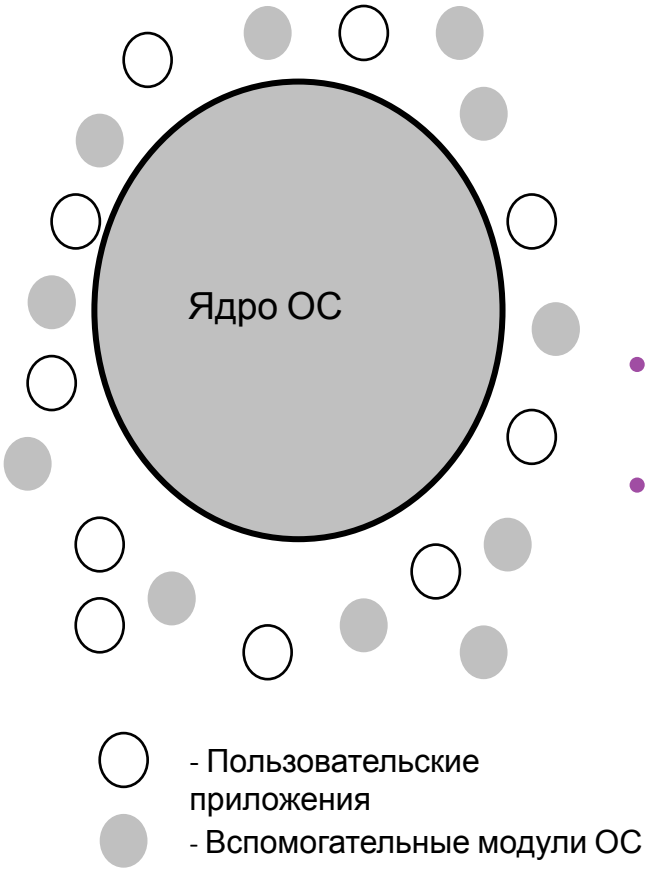
Ядро и вспомогательные модули операционной системы

- Обобщенная структура ОС подразумевает разделение всех ее модулей на две группы:
- **ядро** – модули, выполняющие **основные** функции операционной системы;
- **модули**, выполняющие **вспомогательные** функции операционной системы.
- Модули ядра выполняют такие базовые функции ОС, как управление процессами, памятью, устройствами ввода-вывода и т.п. Без ядра ОС является полностью неработоспособной и не сможет выполнить ни одну из своих функции.

Функции, составляющие ядро ОС

- функции решающие **внутрисистемные задачи** по организации вычислительного процесса и недоступные для приложений;
- функции служащие для **поддержки приложений**, создают ***прикладную программную среду***. Приложения могут обращаться к ядру с запросами – системными вызовами – для выполнения тех или иных действий. Эти функции образуют интерфейс прикладного программирования – **API**.
- Функции ядра являются наиболее **часто используемыми**, скорость их выполнения определяет производительность всей системы в целом. Для обеспечения высокой производительности ОС все модули ядра или большая их часть *постоянно находятся* в оперативной памяти компьютера, то есть являются **резидентными.**

Нечёткость границ между ОС и приложениями



- Обычно ядро ОС оформляется в виде программного модуля некоторого **специального формата**, отличающегося от формата пользовательских приложений. Часть компонентов ОС оформляются как обычные приложения(исполняемые модули стандартного формата).
- Четкой грань между ОС и приложениями нет.
- Решение о включении программа в ОС принимает разработчик ОС. Некоторая программа вначале может существовать как пользовательское приложение, а потом стать частью ОС, или наоборот.

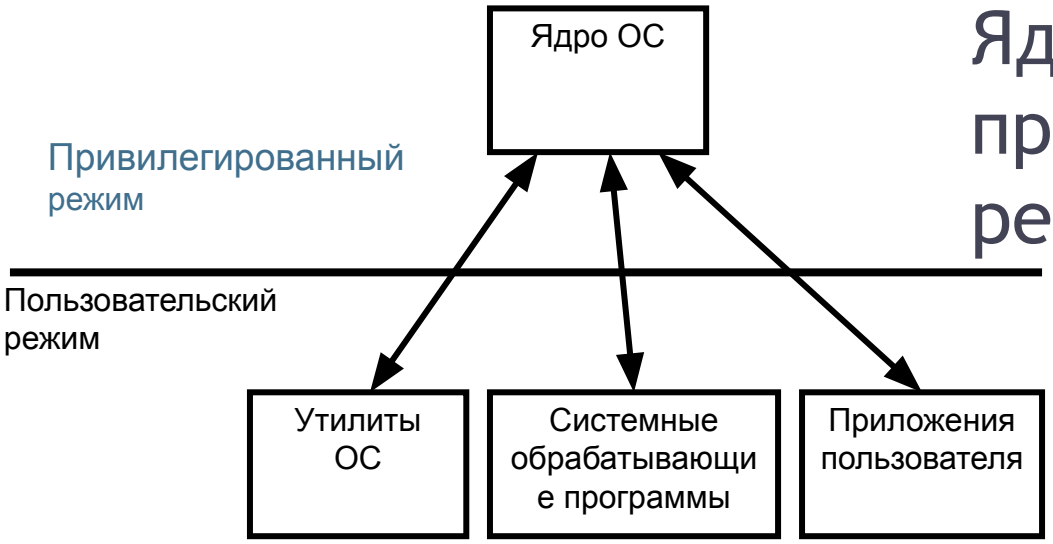
Вспомогательные модули ОС

- ***утилиты*** – программы, решающие отдельные задачи управления и сопровождения компьютерной системы, например программа сжатия дисков и т.п.;
- ***системные обрабатывающие программы*** – текстовые или графические редакторы, компиляторы, компоновщики, отладчики;
- ***программы предоставления пользователю дополнительных услуг*** – специальный вариант пользовательского интерфейса, программа калькулятора, игры;
- ***библиотеки процедур (функций)*** различного назначения, упрощающие разработку приложений, например библиотека математических функций и т.п.
- Для **выполнения своих задач** вспомогательные модули ОС обращаются к функциям ядра посредством **системных вызовов**.
- Вспомогательные модули обычно загружаются в оперативную память компьютера только *на время выполнения* своих функций, то есть являются **транзитными**.

Ядро в привилегированном режиме

- Для надежного управления ходом выполнения приложений операционная система должна иметь по отношению к приложениям определенные привилегии.
- ОС является **арбитром** в споре приложений за **ресурсы** компьютера (получение дополнительной область памяти, время занятия процессора, управление совместно используемыми внешними устройствами)в мультипрограммном режиме.
- Обеспечение привилегии ОС обеспечивается специальными средствами **аппаратной поддержки**.

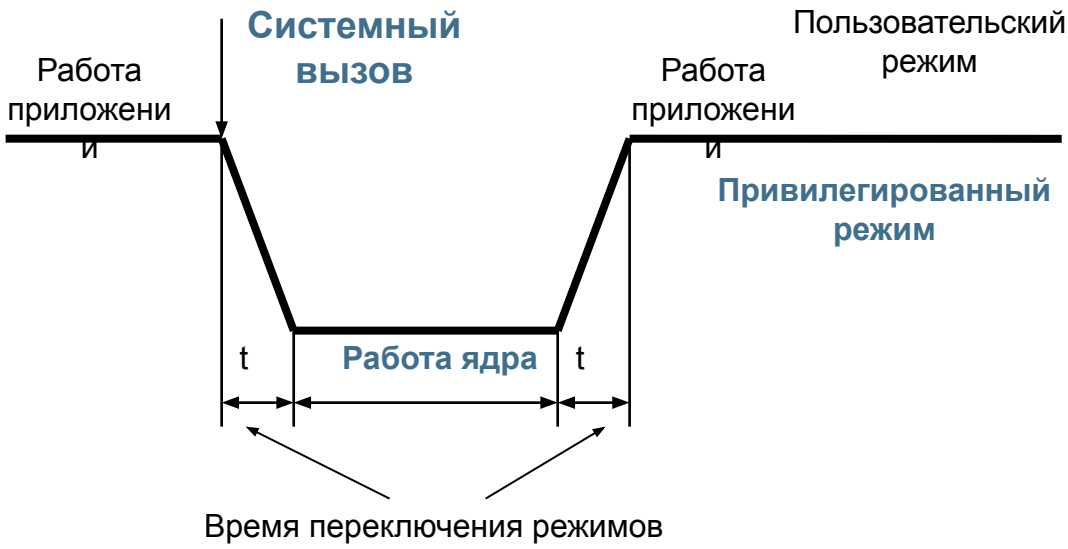
Ядро в привилегированном режиме



- Аппаратура компьютера должна поддерживать как минимум *два режима работы* – **пользовательский** режим ***(user mode)*** и **привилегированный** режим, который также называют ***режимом ядра (kernel mode)***, или ***режим супервизора (supervisor mode)***. ОС или некоторая ее часть работают в *привилегированном* режиме, а приложения – в *пользовательском* режиме.

Запрет выполнения в пользовательском режиме некоторых критичных команд

- Приложения становятся в подчиненное положение за счет *запрета выполнения* в пользовательском режиме некоторых *критичных команд*, связанных с переключением процессора с задачи на задачу, управления устройствами ввода-вывода, доступом к механизму распределения и защиты памяти.
- Выполнение некоторых команд в пользовательском режиме *запрещается безусловно*, другие запрещается выполнять только *при определенных условиях*.
- Условия разрешения выполнения критичных команд находятся под полным контролем ОС , контроль обеспечивается за счет *набора команд*, *безусловно запрещенных* для пользовательского режима.



Повышение устойчивости ОС, обеспечиваемое переходом ядра в привилегированный режим, достигается за счет некоторого замедления выполнения системных вызовов.

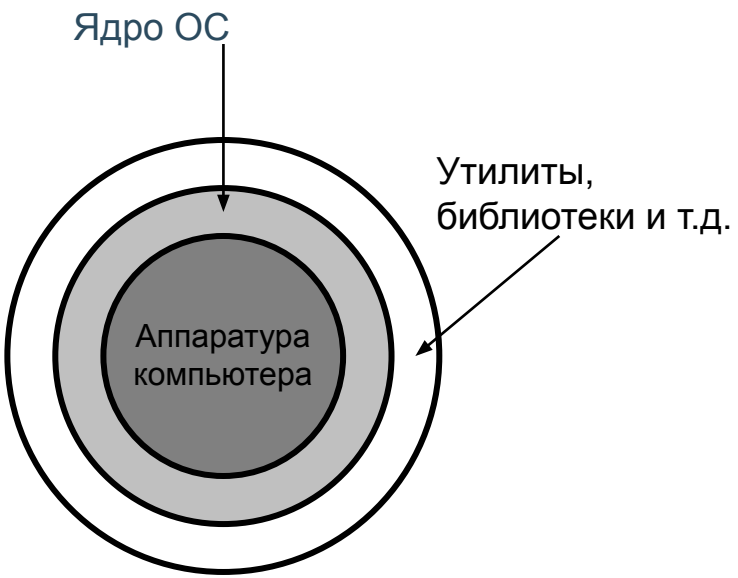
- Системный вызов привилегированного ядра инициирует **переключение процессора** из пользовательского **режима** в привилегированный, а при возврате к приложению – переключение из привилегированного режима в пользовательский.
- Во всех типах процессоров из-за дополнительной *двукратной задержки переключения* переход на процедуру со сменой режима, выполняются медленнее, чем вызов процедуры без смены режима.
- В некоторых случаях, для повышения быстродействия, разработчики ОС отступают от этого классического варианта архитектуры, организуя работу ядра и приложений **в одном и том же режиме**. В этом случае снижается надежность ОС, которая компенсируется за счет тщательной отладки каждого приложения.

Однорежимные ОС

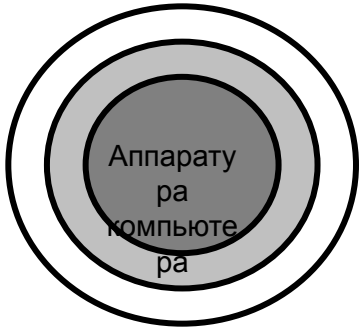
- В одном режиме работают ядро и приложения тех ОС, которые разработаны для процессоров, **не поддерживающих привилегированного режима** работы.
- Наиболее популярным процессором такого типа был процессор Intel 8088/86. MS DOS состояла из двух модулей msdos.sys и io.sys, составляющих ядро системы, к которым с системными вызовами обращались командный интерпретатор **command.com**, системные утилиты и приложения. Некорректно написанные приложения могли разрушить основные модули MS DOS.
- Возможности работать в привилегированном режиме не было использовано разработчиками MS DOS .
- Эта ОС всегда работала на процессорах данного типа в так называемом ***реальном режиме***, в котором эмулируется процессор 8088/86.
- Реальный режим был реализован только для совместимости поздних моделей процессоров с ранней моделью 8088/86 и альтернативой ему является ***защищенный режим*** работы процессора, в котором становятся доступны все особенности процессоров поздних моделей.

Многослойная структура ОС

.



- Компьютер, работающий под управлением ОС на основе ядра, можно рассматривать как систему, состоящую из трех иерархически расположенных слоев: нижний слой - аппаратура, промежуточный –ядро, верхний слой системы – утилиты



Многослойная структура ОС

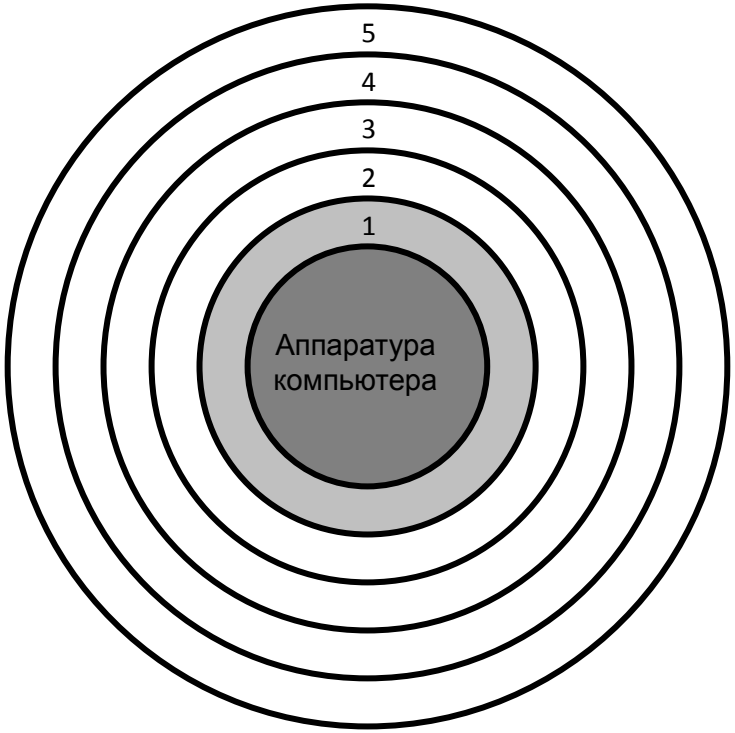
.

- Каждый слой может *взаимодействовать* только со *смежными* слоями. Каждый слой *обслуживает вышележащий слой*, выполняя для него некоторый набор функций, которые образуют межслойный интерфейс.
- На основе функций нижележащего слоя следующий (верхний по иерархии) слой строит свои функции – более сложные и более мощные, которые, в свою очередь, оказываются примитивами для создания еще более мощных функций вышележащего слоя.

Многослойная структура ОС

- Строгие *правила* касаются только *взаимодействия между слоями системы*, а между модулями внутри слоя связи могут быть *произвольными*. Отдельный модуль может выполнить свою работу либо самостоятельно, либо обратиться к другому модулю своего слоя, либо обратиться за помощью к нижележащему слою через межслойный интерфейс.
- Такая организация имеет ряд достоинств. Она существенно упрощает разработку системы, так как позволяет сначала определить “сверху вниз” функции слоев и межслойные интерфейсы, а затем при детальной реализации постепенно наращивать мощность функций слоев, двигаясь “снизу вверх”.
- при модернизации системы можно изменять модули внутри слоя без необходимости производить какие-либо изменения в остальных слоях, если при этих внутренних изменениях межслойные интерфейсы остались в силе.

Слои ядра ОС



1 – Средства аппаратной поддержки ОС
2 – Машино – зависимые модули
3 – Базовые механизмы ядра
4 – Менеджеры ресурсов
5 – Интерфейс системных вызовов

- ***Средства аппаратной поддержки операционной системы.*** До сих пор об операционной системе говорилось как о комплексе программ, хотя часть функций операционной системы может выполняться и аппаратными средствами. В этом случае речь идет не о всей аппаратуре компьютера, а только об аппаратуре непосредственно участвующей в организации вычислительного процесса. Например, средства поддержки привилегированного режима, система прерываний, средства защиты памяти и т.п.
- ***Машино – зависимые компоненты операционной системы.*** Этот слой образуют программные модули, в которых отражается специфика аппаратной платформы компьютера. За счет этого слоя происходит полное экранирование вышележащих слоев ядра от особенностей аппаратной реализации. Тем самым позволяя разрабатывать вышележащие слои на основе Машино - независимых модулей.

Слои ядра ОС

- ***Базовые механизмы ядра.*** Этот слой выполняет наиболее примитивные операции ядра. Модули данного слоя не принимают решений о распределении ресурсов – они только отрабатывают принятые “наверху” решения, что и дает повод называть их исполнительными механизмами для модулей верхних слоев.
- ***Менеджеры ресурсов.*** Этот слой состоит из мощных функциональных модулей, реализующих стратегические задачи по управлению основными ресурсами компьютера. Обычно на данном слое работают менеджеры (диспетчеры) процессов, ввода-вывода, файловой системы и оперативной памяти.
- ***Интерфейс системных вызовов.*** Этот слой является самым верхним слоем ядра и взаимодействует непосредственно с приложениями и системными утилитами, образуя прикладной программный интерфейс операционной системы.

Типовые средства аппаратной поддержки ОС

- Практически все аппаратные платформы имеют некоторый типичный набор средств аппаратной поддержки ОС, в который входят следующие компоненты:
- средства поддержки привилегированного режима;
- средства трансляции адресов;
- средства переключения процессов;
- система прерываний;
- системный таймер;
- средства защиты областей памяти.

Типовые средства аппаратной поддержки ОС

- ***Средства поддержки привилегированного режима*** обычно основаны **на *системном регистре процессора***, часто называемом «словом состояния» машины или процессора. Смена режима привилегий выполняется за счет изменения слова состояния в результате *прерывания* или выполнения *привилегированной команды*.
- В обязанности средств поддержки привилегированного режима входит выполнение *проверки допустимости* выполнения активной программой *инструкций процессора* при текущем уровне привилегированности.

Типовые средства аппаратной поддержки ОС

- ***Средства трансляции адресов*** выполняют операции преобразования виртуальных адресов, которые содержатся в кодах процесса, в адреса физической памяти.
- ***Средства переключения процессов*** предназначены для быстрого *сохранения контекста* приостанавливаемого процесса и *восстановления контекста* процесса, который становится активным. Содержимое контекста обычно включает содержимое всех регистров общего назначения процессора, регистра флагов операций (то есть флагов нуля, переноса, переполнения и т. п.), а также тех системных регистров и указателей, которые связаны с отдельным процессом, а не ОС.

Типовые средства аппаратной поддержки ОС

- ***Система прерываний*** позволяет компьютеру реагировать на внешние события, синхронизировать выполнение процессов и работу устройств ввода-вывода, быстро переходить с одной программы на другую. Механизм прерываний нужен для того, чтобы оповестить процессор о возникновении в вычислительной системе некоторого непредсказуемого события или события, которое не синхронизировано с циклом работы процессора.

Типовые средства аппаратной поддержки ОС

- ***Системный таймер*** часто реализуется в виде быстродействующего регистра-счетчика, необходим ОС для выдержки интервалов времени. По истечению требуемого интервала времени таймер инициирует *прерывание*, которое обрабатывается процедурой ОС. Прерывания от системного таймера используются ОС в первую очередь для слежения за тем, как отдельные процессы *расходуют время процессора*.
- ***Средства защиты областей памяти*** обеспечивают на аппаратном уровне проверку возможности программного кода осуществлять с данными определенной области памяти такие операции, как чтение, запись или выполнение (при передачах управления).

Машино - зависимые компоненты ОС

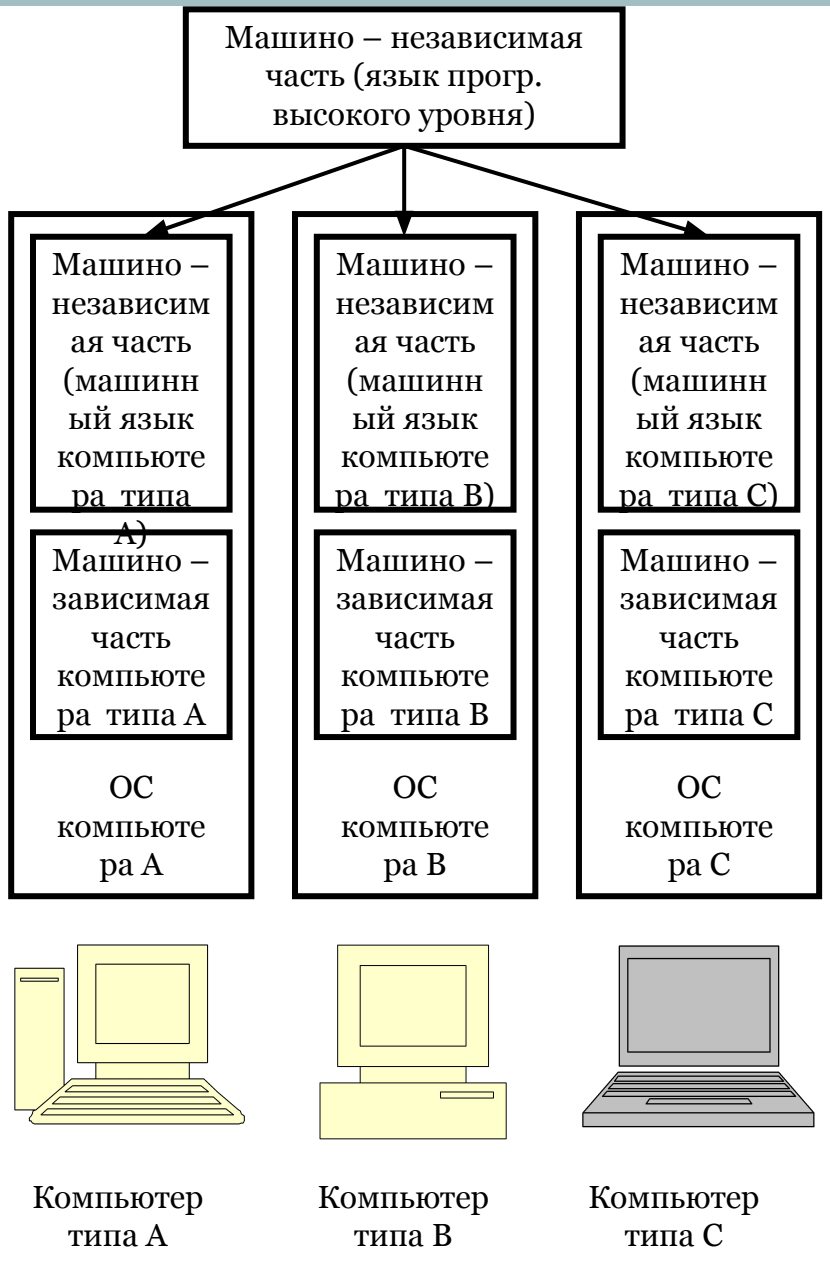
- Одна и та же ОС не может без каких-либо изменений устанавливаться на компьютерах, отличающихся типом процессор или/и способом организации всей аппаратуры.
- ядро можно спроектировать таким образом, что только часть модулей будут машинно - зависимыми, а остальные не будут зависеть от особенностей аппаратной платформы.
- Объем машинно - зависимых компонентов ОС зависит величины отличия в аппаратных платформах, для которых разрабатывается операционная система.
- Несовпадение системы команд процессоров – преодолевается достаточно просто. ОС программируется на языке высокого уровн а затем соответствующим компилятором вырабатывается код для конкретного типа процессора.
- Производители ОС ограничивают распространение своей ОС только несколькими типами процессоров и созданными на их базе аппаратными платформами.

Машино - зависимые компоненты ОС

- Для компьютеров на основе процессоров Intel x86/Pentium разработка экранирующего машинно - зависимого слоя упрощается за счет встроенной в постоянную память компьютера базовой системы ввода-вывода - BIOS.
- **BIOS** содержит *драйверы* для всех устройств, входящих в *базовую конфигурацию компьютера*: жестких и гибких дисков, клавиатуры, дисплея и т.д. Эти драйверы выполняют примитивные операции по управлению устройствами компьютера, за счет этих операций *экранируются различия* аппаратных платформ созданных на процессорах фирмы Intel или совместимых с ними процессоров разными производителями. Разработчики ОС могут пользоваться слоем драйверов BIOS как частью машинно - зависимого слоя ОС, а могут и заменить все или часть драйверов BIOS компонентами ОС.

Переносимость ОС

- Если код ОС может быть сравнительно легко перенесен с процессора одного типа на процессор другого типа и с аппаратной платформы одного типа на аппаратную платформу другого типа, то такую ОС называют ***переносимой*** или ***мобильной.***
- ***Правила*** для обеспечения свойства мобильности ОС:
  - Большая часть кода ОС должна быть написана на языке, *трансляторы* которого *имеются на всех компьютерах*, куда предполагается перенести систему. Стандартные языки высокого уровня (язык Си).
  - *Объем машинно - зависимых частей кода*, которые непосредственно взаимодействуют с аппаратными средствами, должен быть по возможности *минимизирован*.
  - *Аппаратно- зависимый код* должен быть надежно *изолирован в нескольких модулях*, а не быть распределен по всей ОС.



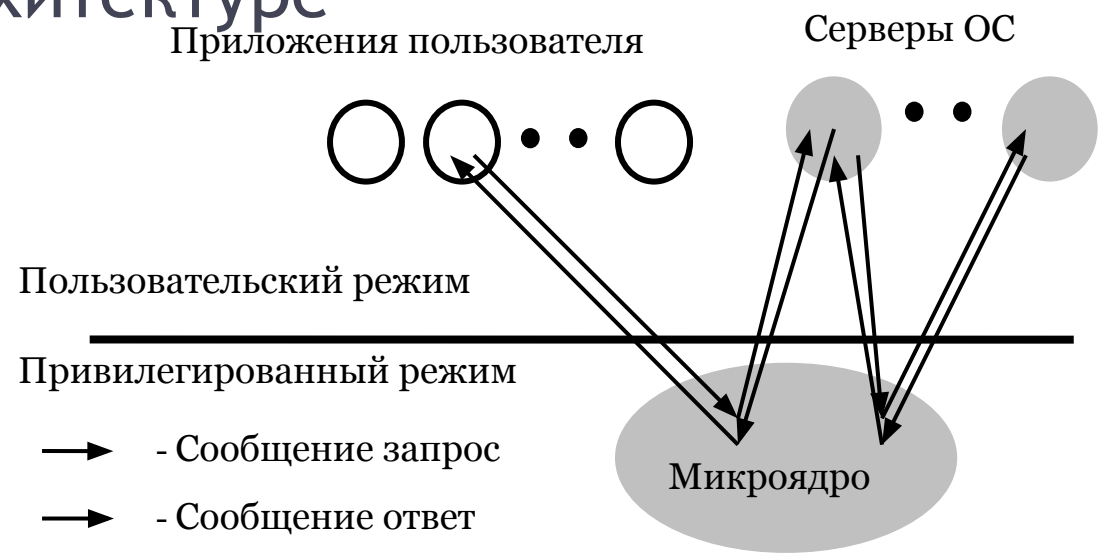
Перенос ОС на разные аппаратные платформы

- В идеале, если реальную аппаратуру подменить неким унифицированным виртуальным компьютером, одинаковым для всех вариантов аппаратной платформы, то все слои ОС выше слоя машинно - зависимых компонентов, могут быть написаны для управления этой виртуальной аппаратурой. Уразработчиков появляется возможность создавать один вариант машинно - независимой части ОС (ядра, утилиты, системные обрабатывающие программы) для всего набора поддерживаемых платформ.

Микроядерная архитектура

- При микроядерной архитектуре в привилегированном режиме остается работать только очень небольшая часть ОС, называемая **микроядром.**
- Микроядро защищено от остальных частей ОС и приложений. В *состав микроядра* обычно входят *машинно - зависимые* модули, модули выполняющие *базовые функции ядра*. Все остальные базовые функции и другие, более высокоуровневые функции ядра оформляются в виде приложений, работающих в пользовательском режиме.

Реализация системных вызовов в микроядерной архитектуре



- Клиент запрашивает выполнение некоторой функции у соответствующего сервера, посылая ему сообщение. Адресные пространства приложений изолированы. Передача сообщений осуществляется через микроядро, выполняющееся в привилегированном режиме (имеет доступ к адресным пространствам каждого из этих приложений). Микроядро сначала передает сообщение (имя и параметры вызываемой процедуры) нужному серверу, сервер выполняет запрошенную операцию, после чего ядро возвращает результат клиенту с помощью другого сообщения. Таким образом, работа микроядерной ОС соответствует модели клиент- сервер, в которой роль транспортных средств выполняет микроядро.